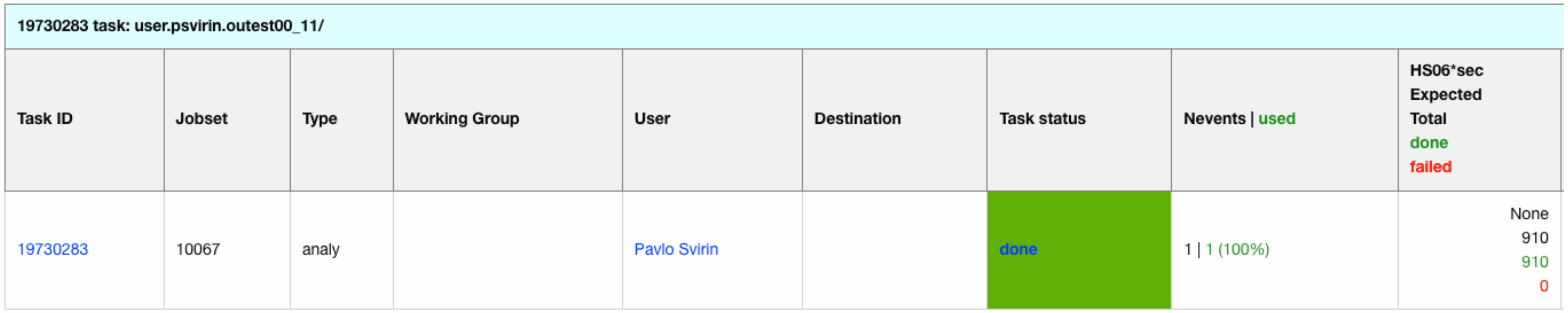
PanDA/Pegasus integration: a case study

Pavlo Svirin

Problem

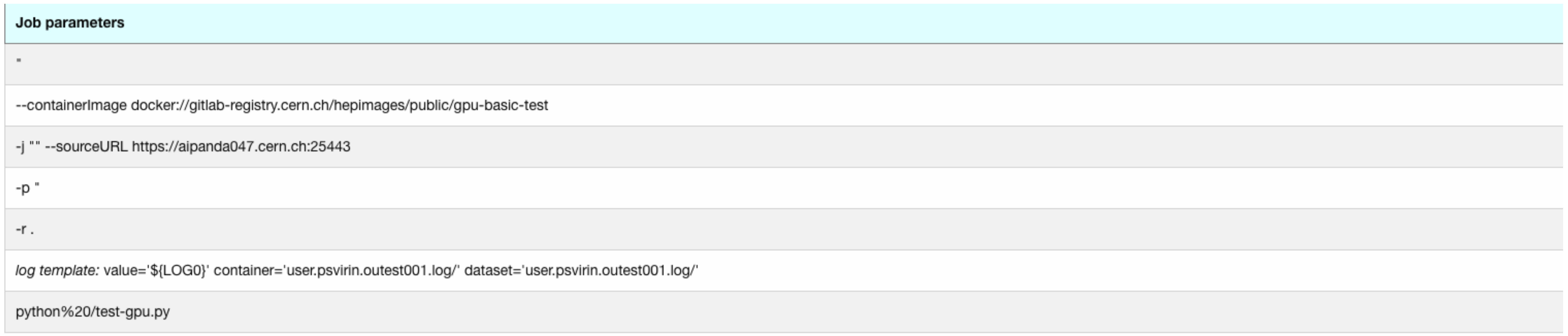
- In order to perform analysis users submit set of tasks
- It is possible to specify independent tasks and tasks with one parent task
- How to specify complex workflows with more than one parents?

Analysis tasks

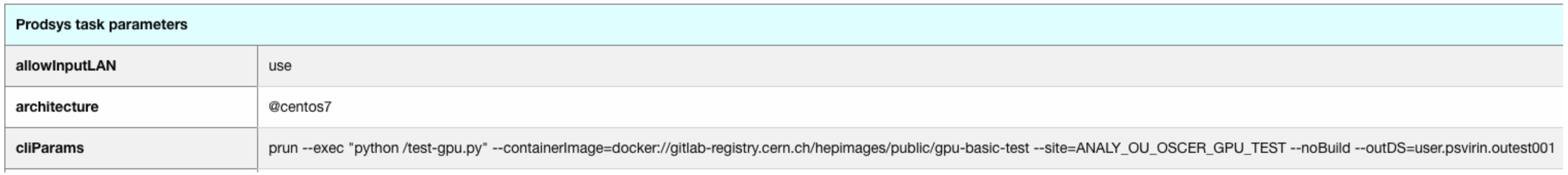


19730283 task: user.psvirin.outest00_11/

Task ID	Jobset	Type	Working Group	User	Destination	Task status	Nevents | used	HS06*sec Expected Total done failed
19730283	10067	analy		Pavlo Svirin		done	1 | 1 (100%)	None 910 910 0



Job parameters
'
--containerImage docker://gitlab-registry.cern.ch/hepimages/public/gpu-basic-test
-j "" --sourceURL https://aipanda047.cern.ch:25443
-p "
-r .
log template: value='${LOG0}' container='user.psvirin.outest001.log/' dataset='user.psvirin.outest001.log/'
python%20/test-gpu.py



Prodsys task parameters	
allowInputLAN	use
architecture	@centos7
cliParams	prun --exec "python /test-gpu.py" --containerImage=docker://gitlab-registry.cern.ch/hepimages/public/gpu-basic-test --site=ANALY_OU_OSCER_GPU_TEST --noBuild --outDS=user.psvirin.outest001

Pegasus WFMS

Developed since 2001 by ISI at the University of Southern California (USC)

Portability across heterogeneous infrastructure
- Separation of workflow description and execution
- Support for campus and leadership class clusters, OSG, XSEDE, academic and commercial clouds
- Can interact with a number of different storage systems (with different protocols)

Supports data reuse – useful in collaborations and ensemble workflow runs

Reliability
- Recovers from failures, retry, workflow-level checkpointing

Scalability
- O(million) task, O(TB) data in a workflow

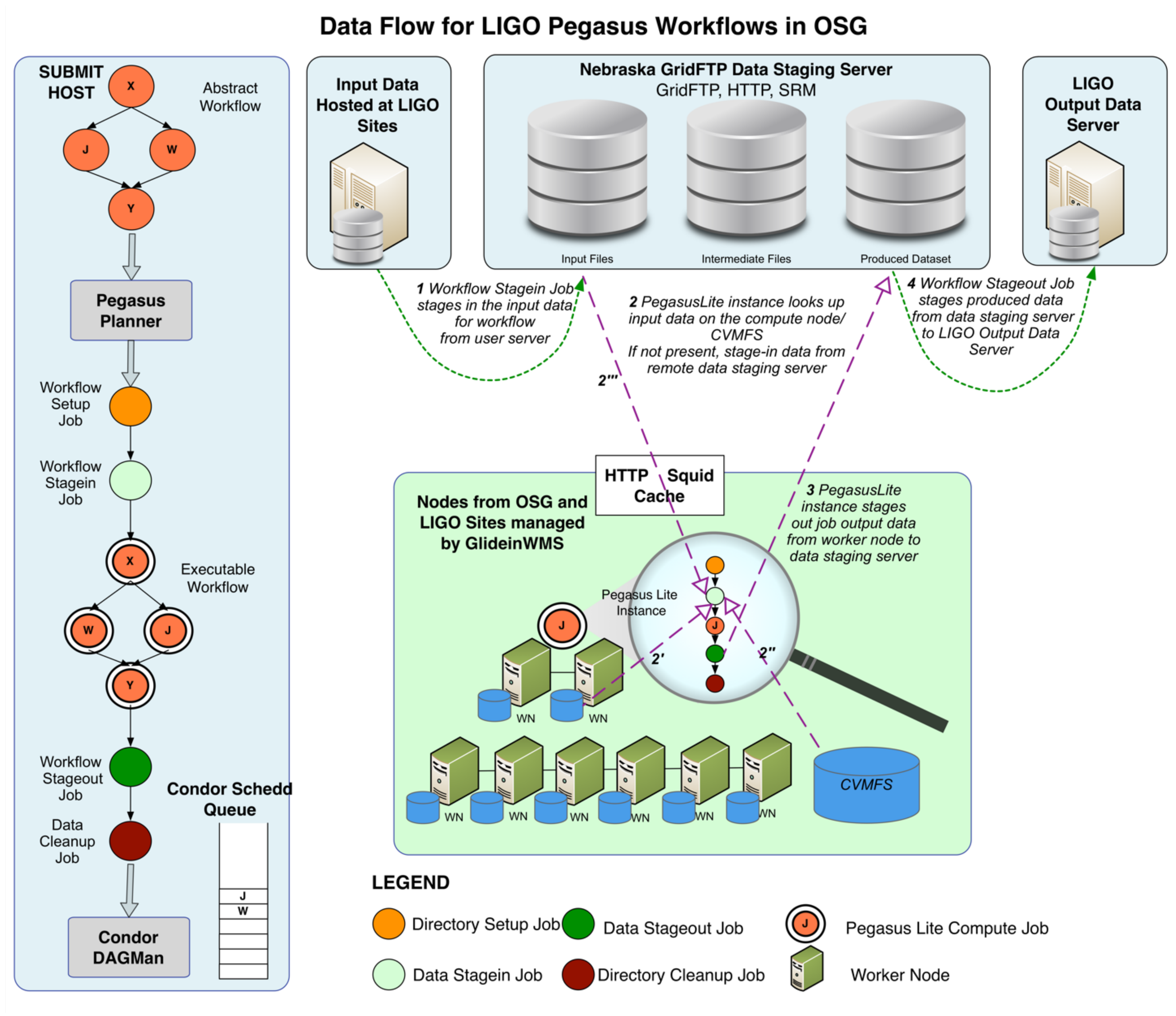
Restructures workflow for performance

Support reproducibility

Web-based monitoring and debugging tools

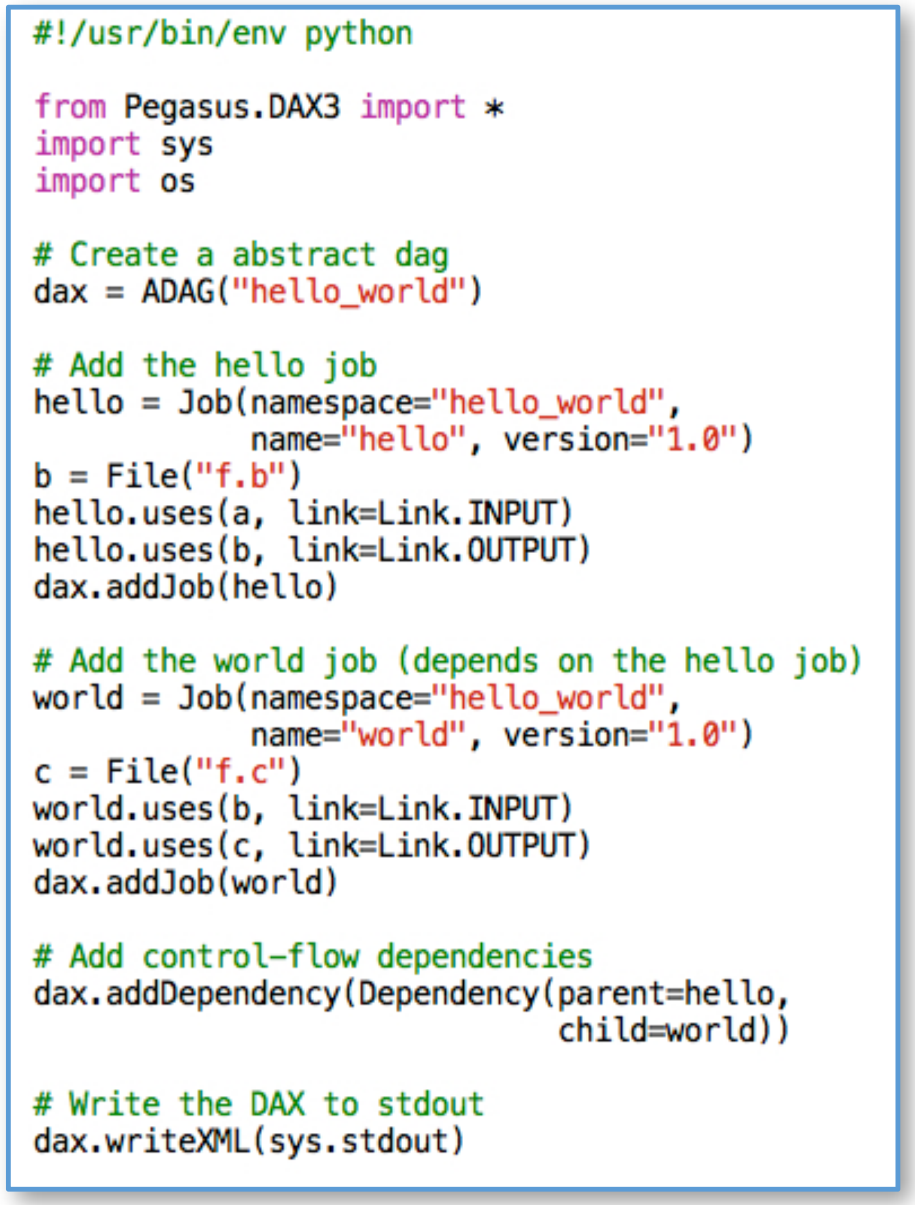
Can be included in various user-facing infrastructures
- Graphical composition tools
- Portals, HUBZero

Pegasus architecture

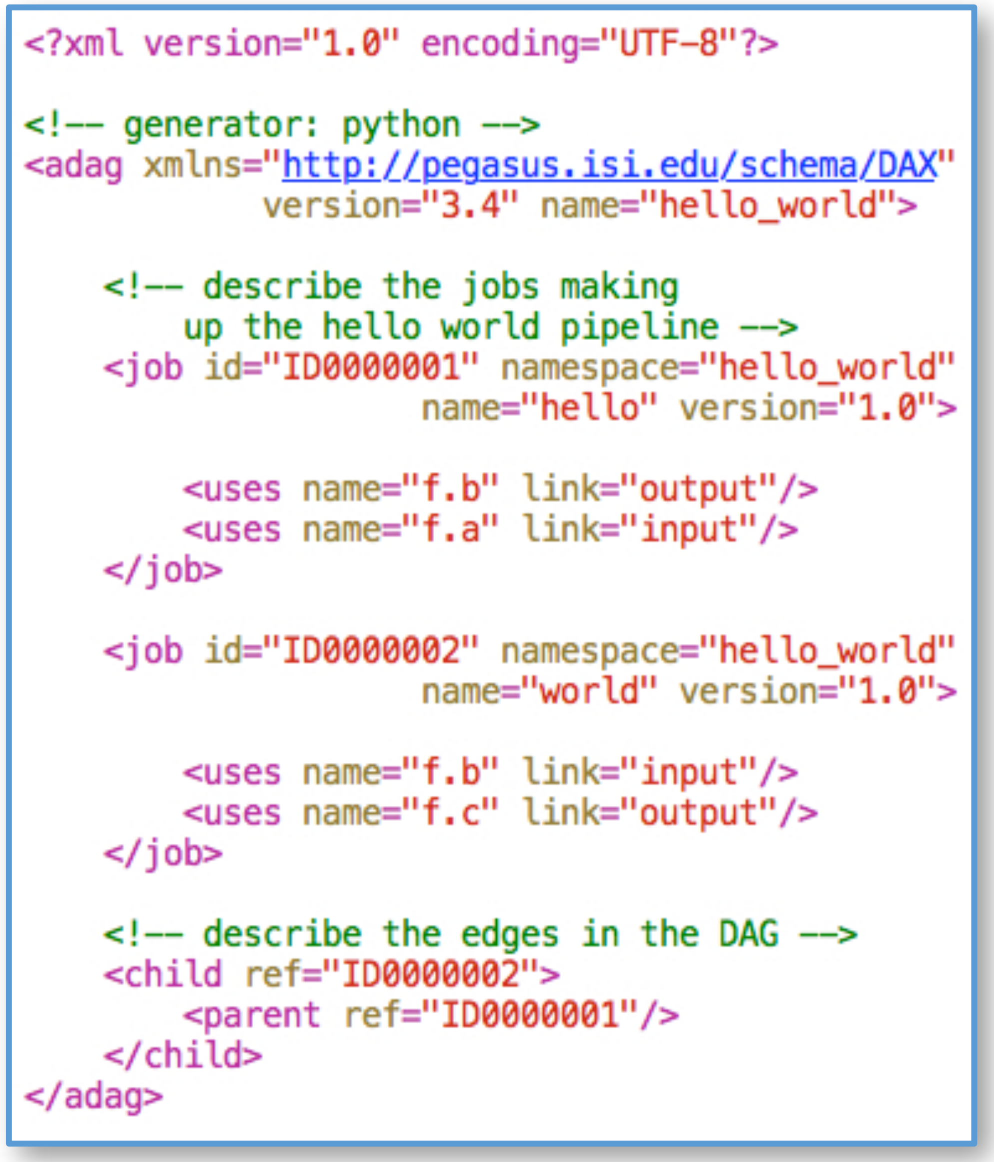


Data Flow for LIGO Pegasus Workflows in OSG
SUBMIT HOST
Abstract Workflow
X
J
W
Y
Pegasus Planner
Workflow Setup Job
Workflow Stagein Job
Executable Workflow
Workflow Stageout Job
Data Cleanup Job
Condor Schedd Queue
Condor DAGMan
Input Data Hosted at LIGO Sites
Nebraska GridFTP Data Staging Server
GridFTP, HTTP, SRM
Input Files
Intermediate Files
Produced Dataset
LIGO Output Data Server
1 Workflow Stagein Job stages in the input data for workflow from user server
2 PegasusLite instance looks up input data on the compute node/ CVMFS If not present, stage-in data from remote data staging server
4 Workflow Stageout Job stages produced data from data staging server to LIGO Output Data Server
2'''
HTTP Squid Cache
Nodes from OSG and LIGO Sites managed by GlideinWMS
3 PegasusLite instance stages out job output data from worker node to data staging server
Pegasus Lite Instance
2'
2''
WN
CVMFS
LEGEND
Directory Setup Job
Data Stageout Job
Pegasus Lite Compute Job
Data Stagein Job
Directory Cleanup Job
Worker Node

Abstract Workflows (DAX)



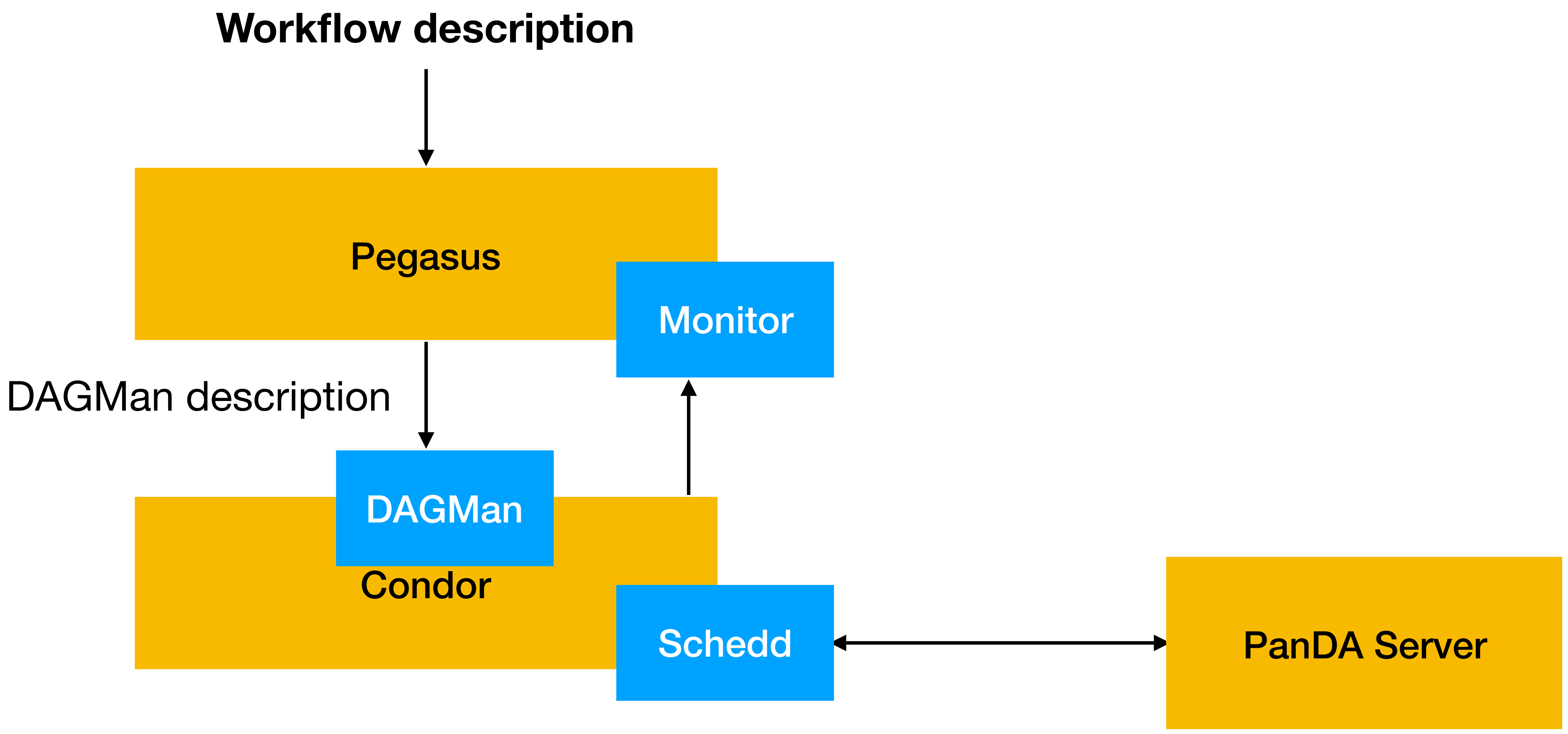
```python
#!/usr/bin/env python

from Pegasus.DAX3 import *
import sys
import os

# Create a abstract dag
dax = ADAG("hello_world")

# Add the hello job
hello = Job(namespace="hello_world",
            name="hello", version="1.0")
b = File("f.b")
hello.uses(a, link=Link.INPUT)
hello.uses(b, link=Link.OUTPUT)
dax.addJob(hello)

# Add the world job (depends on the hello job)
world = Job(namespace="hello_world",
            name="world", version="1.0")
c = File("f.c")
world.uses(b, link=Link.INPUT)
world.uses(c, link=Link.OUTPUT)
dax.addJob(world)

# Add control-flow dependencies
dax.addDependency(Dependency(parent=hello,
                             child=world))

# Write the DAX to stdout
dax.writeXML(sys.stdout)
```



```xml
<?xml version="1.0" encoding="UTF-8"?>

<!-- generator: python -->
<adag xmlns="http://pegasus.isi.edu/schema/DAX"
      version="3.4" name="hello_world">

    <!-- describe the jobs making
         up the hello world pipeline -->
    <job id="ID0000001" namespace="hello_world"
         name="hello" version="1.0">

        <uses name="f.b" link="output"/>
        <uses name="f.a" link="input"/>
    </job>

    <job id="ID0000002" namespace="hello_world"
         name="world" version="1.0">

        <uses name="f.b" link="input"/>
        <uses name="f.c" link="output"/>
    </job>

    <!-- describe the edges in the DAG -->
    <child ref="ID0000002">
        <parent ref="ID0000001"/>
    </child>
</adag>
```

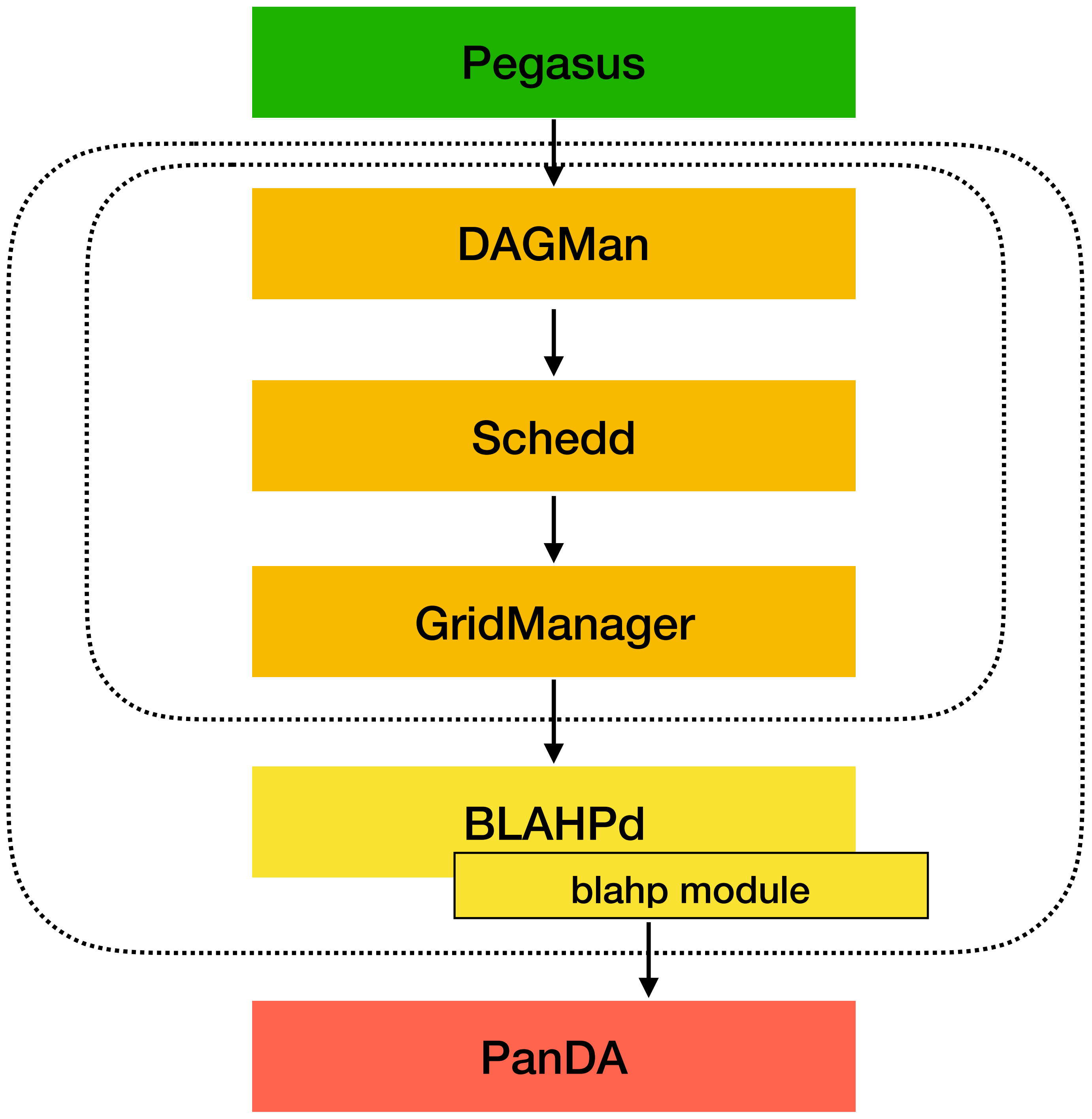
Workflow submission also possible through Jupyter notebooks

Pegasus and PanDA integration



Pegasus and PanDA

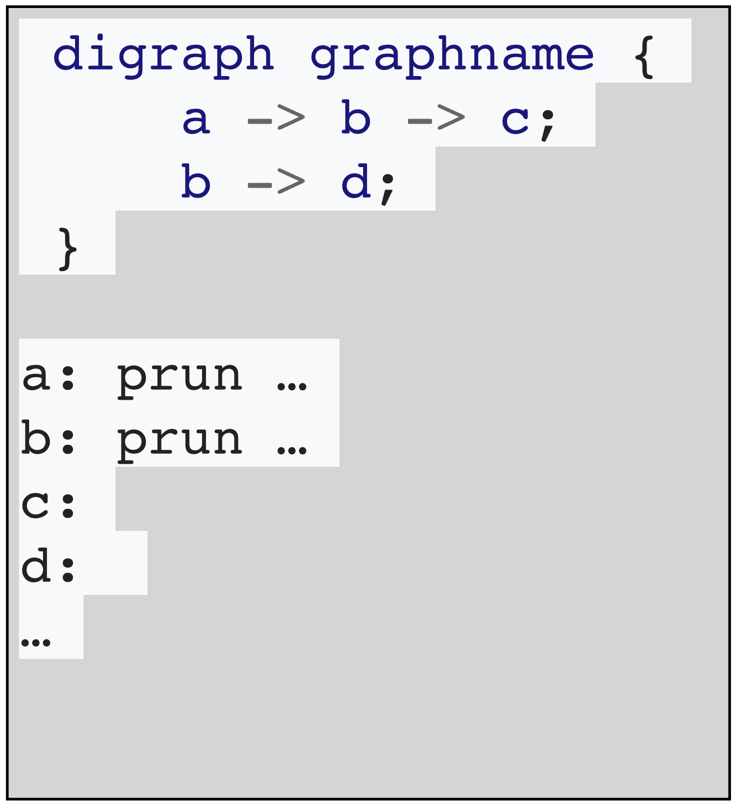
- Integration with PanDA: done in December 2018
- Tested with separate jobs
- Support for tasks added in October, 2019



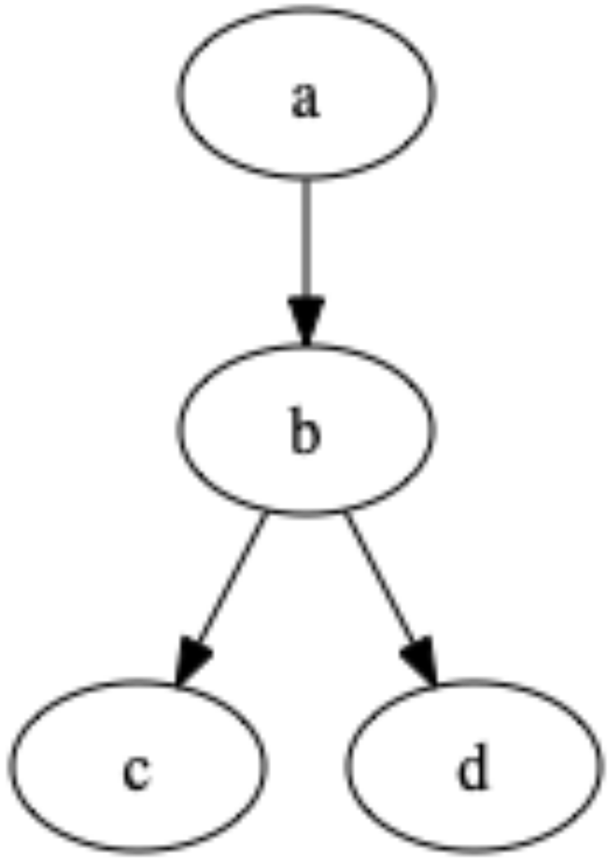
Pegasus: “personal” setup

- Users can run and control workflows
    - from local machine (Pegasus and Condor installation needed)
        - good if input data is on local machine
    - from a virtual machine in CERN OpenStack (an image containing pre-installed Condor and Pegasus can be created)
- user scripts can be created so they do not need to study DAX, just specify a workflow in DOT format
- Problem: proxy expiration

DOT workflow

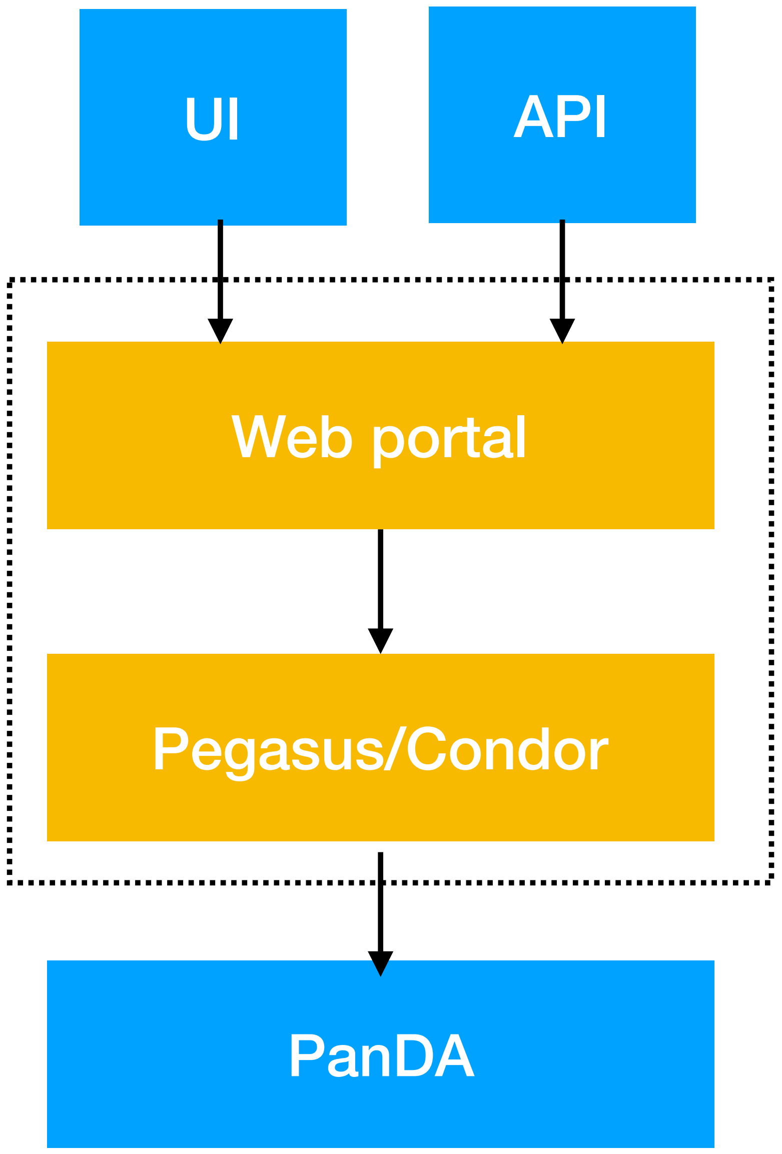


```
 digraph graphname {
        a -> b -> c;
        b -> d;
 }

a: prun …
b: prun …
c:
d:
…
```

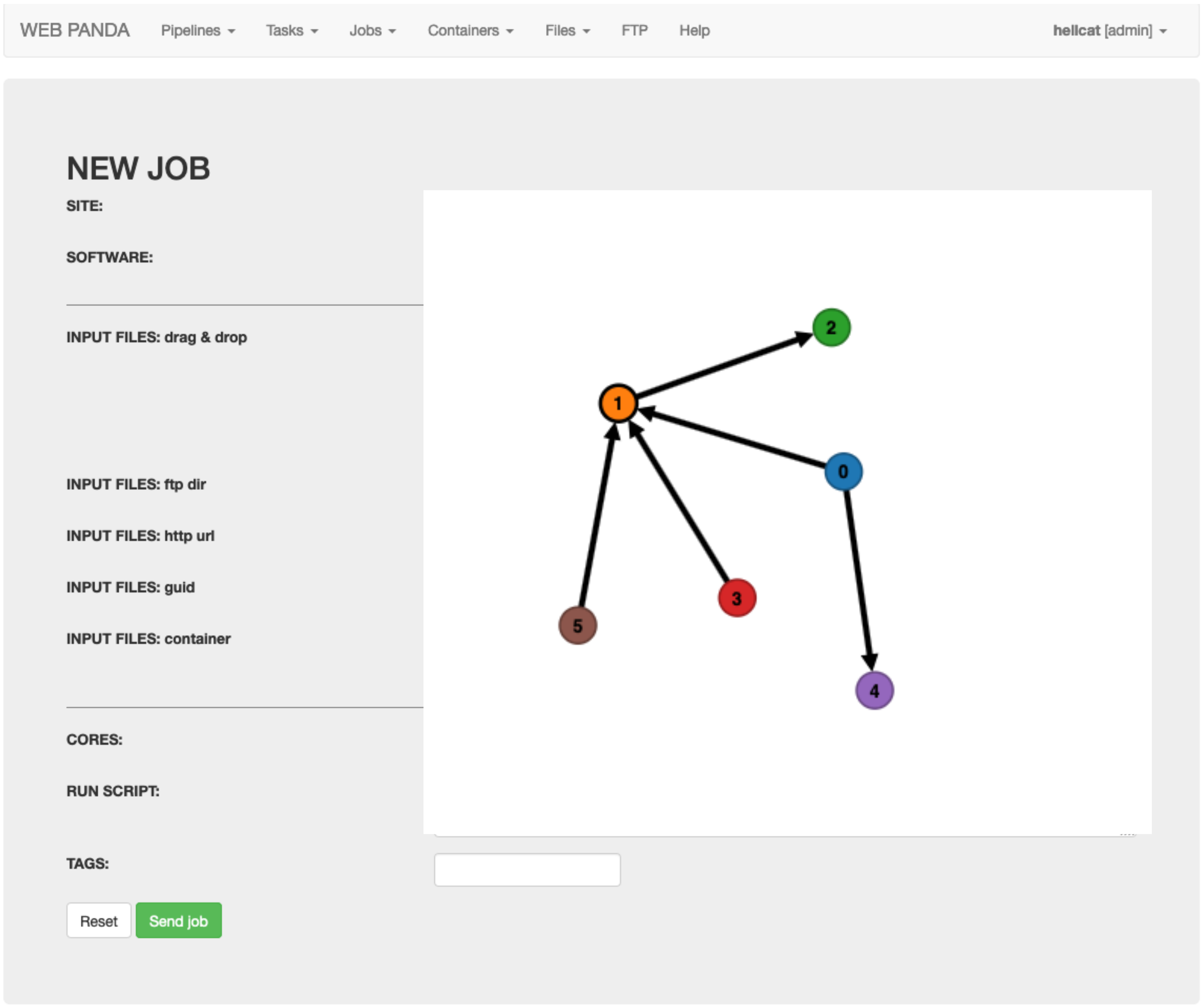


Pegasus as a service

- A portal which can be accessed via CERN SSO
- Users upload DOT files which converted into DAX
- Proxy substitution:
  - to submit tasks under “Workflow Robot” proxy
- Advantage:
  - nothing is needed to install
  - workflow visualisation
  - can run very long workflows without manual intrusion
- Problems: how to transfer local data to dataset?
  - can be solved if data is stored somewhere on AFS



Job management: UI sketch



NEW JOB

SITE:

SOFTWARE:

INPUT FILES: drag & drop

INPUT FILES: ftp dir

INPUT FILES: http url

INPUT FILES: guid

INPUT FILES: container

CORES:

RUN SCRIPT:

TAGS:

UI: by Ivan Tertychnyy, NRC KI

DAG visualisation: by Ross Kirsling

http://bl.ocks.org/rkirsling/5001347

Conclusions and future work

- It is possible to build complex user workflows consisting of tasks and separate jobs
- Todo and possible ways:
  - gather use-cases from users
  - implement easy client tools for fast workflow generation
  - create a pre-configured VM image with pre-installed Pegasus/Condor software
  - estimate opportunities for a workflow portal